

SSSSSSSS	TTTTTTTT	CCCCCCCC	
SSSSSSSS	TTTTTTTT	CCCCCCCC	
SS	TT	CC	
SS	TT	CC	
SS	TT	CC	
SS	TT	CC	
SSSSSS	TT	CC	
SSSSSS	TT	CC	
	TT	CC	
	TT	CC	
	TT	CC	
	TT	CC	
	TT	CC	
SSSSSSSS	TT	CCCCCCCC	
SSSSSSSS	TT	CCCCCCCC	

LL	IIIIII	SSSSSSSS
LL	IIIIII	SSSSSSSS
LL	II	SS
LL	II	SS
LL	II	SS
LL	II	SS
LL	II	SSSSSS
LL	II	SSSSSS
LL	II	
LL	II	SS
LL	II	SS
LL	II	SS
LL	II	SS
LLLLLLLLLL	IIIIII	SSSSSSSS
LLLLLLLLLL	IIIIII	SSSSSSSS


```
0001 0 %TITLE 'Processes the .SEND TOC command'
0002 0 MODULE STC ( IDENT = 'V04-000'
0003 0 %BLISS32[
0004 0 ADDRESSING_MODE(EXTERNAL=LONG_RELATIVE, NONEXTERNAL=LONG_RELATIVE)
0005 0 ]
0006 0 ) =
0007 1 BEGIN
0008 1
0009 1 *****
0010 1 *
0011 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0012 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0013 1 * ALL RIGHTS RESERVED.
0014 1 *
0015 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0016 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0017 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0018 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0019 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0020 1 * TRANSFERRED.
0021 1 *
0022 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0023 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0024 1 * CORPORATION.
0025 1 *
0026 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0027 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0028 1 *
0029 1 *
0030 1 *****
0031 1
0032 1 ++
0033 1 FACILITY: DSR (Digital Standard RUNOFF) / DSRPLUS
0034 1
0035 1 ABSTRACT: Processes the .SEND TOC command
0036 1
0037 1
0038 1 ENVIRONMENT: Transportable
0039 1
0040 1 AUTHOR: R.W.Friday CREATION DATE: May, 1979
0041 1
```

STC
V04-000

Processes the .SEND TOC command
Revision History

G 7
16-Sep-1984 01:48:37
14-Sep-1984 13:08:11

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[RUNOFF.SRC]STC.BLI;1 Page 2 (2)

```

: 43      0042 1 %SBTTL 'Revision History'
: 44      0043 1
: 45      0044 1   MODIFIED BY:
: 46      0045 1
: 47      0046 1       010   REM00010   Ray Marshall   17-November-1983
: 48      0047 1       Modified the external definition of ATABLE to use the new
: 49      0048 1       macro ATABLE_DEFINITION defined in ATCODE.REQ.
: 50      0049 1
: 51      0050 1       009   KFA00009   Ken Alden     16-Mar-1983
: 52      0051 1       Save/restore related items now enabled for DSR as well.
: 53      0052 1
: 54      0053 1       008   REM00008   Ray Marshall   07-Mar-1983
: 55      0054 1       Global edit of all modules. Updated module names, idents,
: 56      0055 1       copyright dates. Changed require files to BLISS library.
: 57      0056 1
: 58      0057 1  !--
```



```

60      0058 1 %SBTTL 'Module Level Declarations'
61      0059 1
62      0060 1 TABLE OF CONTENTS:
63      0061 1
64      0062 1
65      0063 1 INCLUDE FILES:
66      0064 1
67      0065 1 LIBRARY 'NXPORT:XPORT';      ! XPORT Library
68      0066 1 REQUIRE 'REQ:RNODEF';      ! RUNOFF variant definitions
69      0197 1
70      U 0198 1 %IF DSRPLUS %THEN
71      U 0199 1 LIBRARY 'REQ:DPLLIB';      ! DSRPLUS BLISS Library
72      0200 1 %ELSE
73      0201 1 LIBRARY 'REQ:DSRLIB';      ! DSR BLISS Library
74      0202 1 %FI
75      0203 1
76      0204 1 MACROS:
77      0205 1
78      0206 1
79      0207 1 EQUATED SYMBOLS:
80      0208 1
81      0209 1
82      0210 1 EXTERNAL LITERAL
83      0211 1 RINTES : UNSIGNED (8);
84      0212 1
85      0213 1
86      0214 1 OWN STORAGE:
87      0215 1
88      0216 1 OWN
89      0217 1 PP_SCA : $H_R_SCA_BLOCK;      !Used in PUSH_SCA, POP_SCA macros (defined in SCA.REQ).
90      0218 1
91      0219 1 EXTERNAL REFERENCES:
92      0220 1
93      0221 1
94      0222 1 EXTERNAL
95      0223 1 atable : atable_definition, ! Action table. Used to identify what type of
96      0224 1                                     ! action is to be taken on encountering any
97      0225 1                                     ! given character.
98      0226 1
99      0227 1 FLGT : FLAG TABLE,
100     0228 1 GCA : GCA_DEFINITION,
101     0229 1 IRA : FIXED_STRING,
102     0230 1 MRA : REF_FIXED_STRING,
103     0231 1 NPAGEN : PAGE_DEFINITION,
104     0232 1 NUMPRM : NUMPRM_DEFINE,
105     0233 1 PAGEN : PAGE_DEFINITION,
106     0234 1 PHAN : PHAN_DEFINITION,
107     0235 1 SCA : SCA_DEFINITION,
108     0236 1 TSF : TSF_DEFINITION;
109     0237 1 EXTERNAL ROUTINE
110     0238 1 ENDWRD,
111     0239 1 OFT,
112     0240 1 OUTLIN,
113     0241 1 PUTTPG,
114     0242 1 RSKIPS,
115     0243 1 SCANT,
116     0244 1 SETCAS,
```

STC
V04-000

Processes the .SEND TOC command
Module Level Declarations

: 117
: 118

0245 1 SKPSEP;
0246 1

I 7
16-Sep-1984 01:48:37
14-Sep-1984 13:08:11

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[RUNOFF.SRC]STC.BLI;1 Page 4 (3)

STD
V04

.....


```
: 120 0247 1 %SBTTL 'STC --'
: 121 0248 1 GLOBAL ROUTINE stc (handler_code, do_toc) : NOVALUE =
: 122 0249 1
: 123 0250 1 ++
: 124 0251 1 FUNCTIONAL DESCRIPTION:
: 125 0252 1
: 126 0253 1 See ABSTRACT, in module header.
: 127 0254 1
: 128 0255 1 FORMAL PARAMETERS:
: 129 0256 1
: 130 0257 1 HANDLER_CODE - A dummy parameter, used for conformance only.
: 131 0258 1 DO_TOC - If FALSE, the scan is done but no toc entry is made.
: 132 0259 1
: 133 0260 1 IMPLICIT INPUTS: None
: 134 0261 1
: 135 0262 1 IMPLICIT OUTPUTS: None
: 136 0263 1
: 137 0264 1 ROUTINE VALUE:
: 138 0265 1 COMPLETION CODES: None
: 139 0266 1
: 140 0267 1 SIDE EFFECTS: None
: 141 0268 1
: 142 0269 1 --
: 143 0270 2 BEGIN
: 144 0271 2
: 145 0272 2 LOCAL
: 146 0273 2 sca_para_copy,
: 147 0274 2 sca_indent_copy,
: 148 0275 2 hold_ind_flag,
: 149 0276 2 hold_ind_eflag,
: 150 0277 2 hold_all_flags, !Vers. 1(123), KAD.
: 151 0278 2 sca_copy : sca_definition; !Used to preserve SCA.
: 152 0279 2
: 153 0280 2 LOCAL
: 154 0281 2 mra_temp : VECTOR [4 + CH$ALLOCATION (1000)], !Temporary MRA area
: 155 0282 2 tsf_temp : VECTOR [tsf_size]; !Temporary TSF
: 156 0283 2
: 157 0284 2 LOCAL
: 158 0285 2 mra_address, !Preserve interrupted MRA
: 159 0286 2 tsf_address; !Preserve interrupted TSF
: 160 0287 2
: 161 0288 2 !Skip the comma and spaces, if any, following the numerical parameter, if any.
: 162 0289 2 skipsep (ira);
: 163 0290 2
: 164 0291 2 push_sca; !Save the SAVED SCA bits.
: 165 0292 2
: 166 0293 2 INCR i FROM 0 TO sca_size - 1 DO
: 167 0294 2 sca_copy [i] = .sca [i]; !Save SCA
: 168 0295 2
: 169 0296 2 !Initialize SCA prior to scanning the string.
: 170 0297 2 sca_lm = 0;
: 171 0298 2 sca_rm = 150; !Set wide right margin so won't start a new line.
: 172 0299 2 sca_spacing = 1; !Single spacing avoids spurious intermediate code.
: 173 0300 2 setcas (leave_case); !Force first character of first word
: 174 0301 2 !to upper case, rest to lower case.
: 175 0302 2 sca_fc = true; !This is the first character for the MRA.
: 176 0303 2 sca_fc_case = true; !Use first character-of-word case rules.
```

STC
V04-000

Processes the .SEND TOC command
STC --

K 7
16-Sep-1984 01:48:37
14-Sep-1984 13:08:11

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[RUNOFF.SRC]STC.BLI;1 Page 6
(4)

```

177 0304 2 sca_prescan = false;
178 0305 2 sca_para_copy = .sca_para_pnd;
179 0306 2 sca_indent_copy = .sca_indent;
180 0307 2 sca_para_pnd = false;
181 0308 2 sca_indent = 0;
182 0309 2 sca_fill = false;
183 0310 2 sca_justify = false;
184 0311 2
185 0312 2 !Preserve the addresses of the structures to which these items refer.
186 0313 2 mra_address = .mra;
187 0314 2 tsf_address = .tsf;
188 0315 2
189 0316 2 !Now switch working buffers, so that text currently being accumulated
190 0317 2 !is not disturbed.
191 0318 2 mra = mra_temp;
192 0319 2 tsf = tsf_temp;
193 0320 2
194 0321 2 !Now manually set up MRA, to look like a FIXED STRING.
195 0322 2 fs_maxsize (mra) = 1000;
196 0323 2 fs_init (mra);
197 0324 2 !And that finishes the initialization.
198 0325 2 !Now initialize TSF for SCANT.
199 0326 2
200 0327 2 INCR i FROM 0 TO tsf_size - 1 DO
201 0328 2     tsf [.i] = 0;
202 0329 2
203 0330 2 tsf_btc = true;
204 0331 2 !This TSF record describes a table of contents entry.
205 0332 2 sca_wrd_int_l = 0;
206 0333 2 sca_wrd_ext_l = 0;
207 0334 2 sca_wrd_iseqn = 0;
208 0335 2 sca_wrd_draft = 0;
209 0336 2 sca_wrd_draft_f = 0;
210 0337 2 sca_wrd_bar = 0;
211 0338 2 sca_wrd_bar_chr = 0;
212 0339 2 sca_wrd_seqn_f = 0;
213 0340 2 sca_wrd_ipagen = 0;
214 0341 2 sca_wrd_footw = 0;
215 0342 2 sca_wrd_f_xtn = 0;
216 0343 2 sca_wrd_l_xtn = 0;
217 0344 2 sca_wrd_lc_pnct = 0;
218 0345 2 sca_wrd_lst_sp = 0;
219 0346 2 sca_wrd_lst_jus = 0;
220 0347 2 sca_wrd_lst_und = 0;
221 0348 2 sca_wrd_pntr = .fs_next (mra);
222 0349 2 sca_wrd_cpend = rintes;
223 0350 2 fs_start (mra) = .sca_wrd_pntr;
224 0351 2 hold_ind_flag = .flgt[ind_flag, flag_character]; !Preserve <INDEX flag> status.
225 0352 2 hold_ind_eflag = .flgt[ind_flag, flag_enabled];
226 0353 2 oft Th_no_flags_inde, ind_flag); !Disable <INDEX flag>
227 0354 2
228 0355 2 !Save status of all flags. We disable all flags before scanning the
229 0356 2 !text to be sent to the .BTC file; therefore RUNOFF flags can be
230 0357 2 !sent as normal characters. (Vers. 1(123), KAD.)
231 0358 2 hold_all_flags = .gca_flag_cmd;
232 0359 2 oft Th_no_flags_all, 0); !Disable all flags.
233 0360 2
```

STD
V04

: R


```
234 0361 2 scant (); ! Now let SCANT process the command.
235 0362 2
236 0363 2 oft (.hold_all_flags, 0); ! Restore saved flags.
237 0364 2
238 0365 2 oft (h_no_flags_inde, ind_flag); !Restore <INDEX flag> to former status.
239 0366 2
240 0367 2 flgt [ind_flag, flag_character] = .hold_ind_flag;
241 0368 2 flgt [ind_flag, flag_enabled] = .hold_ind_eflag;
242 0369 2
243 0370 2 IF .flgt [ind_flag, flag_enabled]
244 0371 2 AND .sca_flags
245 0372 2 THEN
246 0373 2 atable [.flgt [ind_flag, flag_character]] = a_flag;
247 0374 2
248 0375 2 !Return here is with a word.
249 0376 2
250 0377 2 IF .do_toc THEN
251 0378 2 BEGIN
252 0379 2
253 0380 2 IF .sca_wrd_cpend NEQ rintes THEN
254 0381 2 endwrd (false, false, false); !End word, but don't write a space too.
255 0382 2
256 0383 2 tsf_major = maj_send; ! Identify this record as something from the
257 0384 2 .SEND TOC command
258 0385 2 tsf_minor = .num_value; ! Copy the value of 'n' from the .SEND command
259 0386 2 into the minor type code
260 0387 2
261 0388 2 !If at the top of any page, look again to see which page number to output.
262 0389 2 !This code added (from HL.BLI) in V1.126c, ident 004.
263 0390 2 IF .gca_btc AND .phan_top_page THEN
264 0391 2 IF .phan_top_first THEN
265 0392 2
266 0393 2 %IF FLIP %THEN
267 0394 2 puttpg (pagen, flip$k_tocpag)
268 0395 2 ELSE
269 0396 2 puttpg (npagen, flip$k_tocpag);
270 0397 2 %ELSE
271 0398 2 puttpg (pagen, -1)
272 0399 2 ELSE
273 0400 2 puttpg (npagen, -1);
274 0401 2 %FI
275 0402 2 outlin (false); ! Don't disturb justification algorithm
276 0403 2 END;
277 0404 2
278 0405 2 INCR i FROM 0 TO sca_size - 1 DO
279 0406 2 sca [.i] = .sca_copy [.i]; ! Restore SCA
280 0407 2
281 0408 2 pop_sca; ! Restore the SAVED SCA bits.
282 0409 2
283 0410 2 mra = .mra_address; ! Restore previous stati.
284 0411 2 tsf = .tsf_address;
285 0412 2 sca_para_pnd = .sca_para_copy;
286 0413 2 sca_indent = .sca_indent_copy;
287 0414 2 END; !End of STC
```

.TITLE STC Processes the .SEND TOC command

.IDENT \V04-000\

.PSECT \$OWNS\$,NOEXE,2

00000 PP_SCA: .BLKB 48

.EXTRN RINTES, ATABLE, FLGT
.EXTRN GCA, IRA, MRA, NPAGEN
.EXTRN NUMPRM, PAGEN, PHAN
.EXTRN SCA, TSF, ENDWRD
.EXTRN OFT, OUTLIN, PUTTPG
.EXTRN RSKIPS, SCANT, SETCAS
.EXTRN SKPSEP

.PSECT \$CODE\$,NOWRT,2

			OFFC 00000		.ENTRY	STC, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 0248		
	5B	00000000G	EF	9E	00002	MOVAB	FLGT+112, R11		
	5A	00000000'	EF	9E	00009	MOVAB	PP_SCA, R10		
	59	00000000G	EF	9E	00010	MOVAB	SCA+100, R9		
	5E	F9E8	CE	9E	00017	MOVAB	-1560(SP), SP		
		00000000G	EF	9F	0001C	PUSHAB	IRA	: 0289	
00000000G	EF		01	FB	00022	CALLS	#1, SKPSEP		
	6A	00	B9	D0	00029	MOVL	@SCA+100, PP_SCA		
04	AA	04	B9	D0	0002D	MOVL	@SCA+104, PP_SCA+4		
08	AA	08	B9	D0	00032	MOVL	@SCA+108, PP_SCA+8		
0C	AA	0C	B9	D0	00037	MOVL	@SCA+112, PP_SCA+12		
10	AA	10	B9	D0	0003C	MOVL	@SCA+116, PP_SCA+16		
14	AA	14	B9	D0	00041	MOVL	@SCA+120, PP_SCA+20		
18	AA	18	B9	D0	00046	MOVL	@SCA+124, PP_SCA+24		
1C	AA	1C	B9	D0	0004B	MOVL	@SCA+128, PP_SCA+28		
20	AA	20	B9	D0	00050	MOVL	@SCA+132, PP_SCA+32		
24	AA	24	B9	D0	00055	MOVL	@SCA+136, PP_SCA+36		
28	AA	28	B9	D0	0005A	MOVL	@SCA+140, PP_SCA+40		
2C	AA	2C	B9	D0	0005F	MOVL	@SCA+144, PP_SCA+44		
			50	D4	00064	CLRL	I	: 0293	
	FE80	CD40	9C	A940	D0	00066	1\$: MOVL	SCA[I], SCA_COPY[I]	: 0294
FO		50	0000005F	8F	F3	0006E	AOBLEQ	#95, I, 1\$	
			10	B9	D4	00076	CLRL	@SCA+116	: 0297
14	B9	96	8F	9A	00079	MOVZBL	#150, @SCA+120	: 0298	
18	B9		01	D0	0007E	MOVL	#1, @SCA+124	: 0299	
			7E	D4	00082	CLRL	-(SP)	: 0300	
00000000G	EF		01	FB	00084	CALLS	#1, SETCAS		
30	A9		01	D0	0008B	MOVL	#1, SCA+148	: 0302	
6C	A9		01	D0	0008F	MOVL	#1, SCA+208	: 0303	
		48	A9	D4	00093	CLRL	SCA+172	: 0304	
	57	78	A9	7D	00096	MOVQ	SCA+220, SCA_INDENT_COPY	: 0306	
		78	A9	7C	0009A	CLRQ	SCA+220	: 0308	
		04	B9	D4	0009D	CLRL	@SCA+104	: 0309	
		00	B9	D4	000A0	CLRL	@SCA+100	: 0310	
	56	00000000G	EF	D0	000A3	MOVL	MRA, MRA_ADDRESS	: 0313	
	55	00000000G	EF	D0	000AA	MOVL	TSF, TSF_ADDRESS	: 0314	
00000000G	EF	00A0	CE	9E	000B1	MOVAB	MRA_TEMP, MRA	: 0318	
00000000G	EF		6E	9E	000BA	MOVAB	TSF_TEMP, TSF	: 0319	
	50	00000000G	EF	D0	000C1	MOVL	MRA, R0	: 0322	
08	A0	03E8	8F	3C	000C8	MOVZWL	#1000, 8(R0)		
		0C	A0	D4	000CE	CLRL	12(R0)	: 0323	

	04	60	10	A0	9E	000D1	MOVAB	16(R0), (R0)		
		A0		60	D0	000D5	MOVL	(R0), 4(R0)		
		52	00000000G	EF	D0	000D9	MOVL	TSF, R2		0328
				51	D4	000E0	CLRL	I		
				6241	D4	000E2	CLRL	(R2)[I]		
F9		51		27	F3	000E5	AOBLEQ	#39, I, 2\$		
	2C	A2		01	D0	000E9	MOVL	#1, 44(R2)		0330
			0098	C9	7C	000ED	CLRQ	SCA+252		0332
			00A0	C9	7C	000F1	CLRQ	SCA+260		0334
			00A8	C9	D4	000F5	CLRL	SCA+268		0336
00AC		C9		01	8A	000F9	BICB2	#1, SCA+272		0337
			00B0	C9	D4	000FE	CLRL	SCA+276		0338
			00B8	C9	7C	00102	CLRQ	SCA+284		0339
			00C0	C9	7C	00106	CLRQ	SCA+292		0341
			00C8	C9	D4	0010A	CLRL	SCA+300		0343
			00E4	C9	7C	0010E	CLRQ	SCA+328		0344
			00EC	C9	7C	00112	CLRQ	SCA+336		0346
0094		C9	04	A0	D0	00116	MOVL	4(R0), SCA+248		0348
00B4		C9	00G	8F	9A	0011C	MOVZBL	#RINTES, SCA+280		0349
		60	0094	C9	D0	00122	MOVL	SCA+248, (R0)		0350
		54		6B	D0	00127	MOVL	FLGT+112, HOLD_IND_FLAG		0351
		53		B8	AB	0012A	MOVL	FLGT+40, HOLD_IND_EFLAG		0352
				0A	DD	0012E	PUSHL	#10		0353
		7E	84	8F	9A	00130	MOVZBL	#132, -(SP)		
00000000G		EF		02	FB	00134	CALLS	#2, OFT		
		52	00000000G	FF	D0	0013B	MOVL	@GCA+132, HOLD_ALL_FLAGS		0358
				7E	D4	00142	CLRL	-(SP)		0359
		7E	7C	8F	9A	00144	MOVZBL	#124, -(SP)		
00000000G		EF		02	FB	00148	CALLS	#2, OFT		
00000000G		EF		00	FB	0014F	CALLS	#0, SCANT		0361
				7E	D4	00156	CLRL	-(SP)		0363
				52	DD	00158	PUSHL	HOLD_ALL_FLAGS		
00000000G		EF		02	FB	0015A	CALLS	#2, OFT		
				0A	DD	00161	PUSHL	#10		0365
		7E	84	8F	9A	00163	MOVZBL	#132, -(SP)		
00000000G		EF		02	FB	00167	CALLS	#2, OFT		
		6B		54	D0	0016E	MOVL	HOLD_IND_FLAG, FLGT+112		0367
B8		AB		53	D0	00171	MOVL	HOLD_IND_EFLAG, FLGT+40		0368
		10	B8	AB	E9	00175	BLBC	FLGT+40, 3\$		0370
		0C	2C	B9	E9	00179	BLBC	@SCA+144, 3\$		0371
		50	00000000G	EF	9E	0017D	MOVAB	ATABLE, R0		0373
00	BB	40		03	90	00184	MOVB	#3, @FLGT+112[R0]		
		62	08	AC	E9	00189	BLBC	DO TOC, 8\$		0377
00000000G		8F	00B4	C9	D1	0018D	CMPL	SCA+280, #RINTES		0380
				0B	13	00196	BEQL	4\$		
				7E	7C	00198	CLRQ	-(SP)		0381
				7E	D4	0019A	CLRL	-(SP)		
00000000G		EF		03	FB	0019C	CALLS	#3, ENDWRD		
		50	00000000G	EF	D0	001A3	MOVL	TSF, R0		
38		A0		03	D0	001AA	MOVL	#3, 56(R0)		0383
3C		A0	00000000G	EF	D0	001AE	MOVL	NUMPRM+4, 60(R0)		0385
		29	00000000G	EF	E9	001B6	BLBC	GCA+124, 7\$		0390
		22	00000000G	EF	E9	001BD	BLBC	PHAN, 7\$		
		0B	00000000G	EF	E9	001C4	BLBC	PHAN+24, 5\$		0391
		7E	00000000G	01	CE	001CB	MNEGL	#1, -(SP)		0398
				EF	9F	001CE	PUSHAB	PAGEN		
				09	11	001D4	BRB	6\$		

7E	00000000G	01	CE	001D6	5\$:	MNEGL	#1, -(SP)	0400
00000000G	EF	02	9F	001D9	6\$:	PUSHAB	NPAGEN	
00000000G	EF	07E	FB	001DF	7\$:	CALLS	#2, PUTTPG	0402
		01	D4	001E6	8\$:	CLRL	-(SP)	
		50	FB	001E8	9\$:	CALLS	#1, OUTLIN	0405
9C A940	FE80	CD40	D4	001EF		CLRL	I	0406
F0	50	0000005F	D0	001F1		MOVL	SCA_COPY[I], SCA[I]	
00	B9		8F	F3	001F9	AOBLEQ	#95, I, 9\$	
04	B9	04	6A	D0	00201	MOVL	PP_SCA, @SCA+100	
08	B9	08	AA	D0	00205	MOVL	PP_SCA+4, @SCA+104	
0C	B9	0C	AA	D0	0020A	MOVL	PP_SCA+8, @SCA+108	
10	B9	10	AA	D0	0020F	MOVL	PP_SCA+12, @SCA+112	
14	B9	14	AA	D0	00214	MOVL	PP_SCA+16, @SCA+116	
18	B9	18	AA	D0	00219	MOVL	PP_SCA+20, @SCA+120	
1C	B9	1C	AA	D0	0021E	MOVL	PP_SCA+24, @SCA+124	
20	B9	20	AA	D0	00223	MOVL	PP_SCA+28, @SCA+128	
24	B9	24	AA	D0	00228	MOVL	PP_SCA+32, @SCA+132	
28	B9	28	AA	D0	0022D	MOVL	PP_SCA+36, @SCA+136	
2C	B9	2C	AA	D0	00232	MOVL	PP_SCA+40, @SCA+140	
00000000G	EF		AA	D0	00237	MOVL	PP_SCA+44, @SCA+144	
00000000G	EF		56	D0	0023C	MOVL	MRA_ADDRESS, MRA	0410
78	A9		55	D0	00243	MOVL	TSF_ADDRESS, TSF	0411
			57	7D	0024A	MOVQ	SCA_INDENT_COPY, SCA+220	0413
			04	0024E		RET		0414

; Routine Size: 591 bytes, Routine Base: \$CODE\$ + 0000

: 288 0415 1
: 289 0416 1 END
: 290 0417 0 ELUDOM

!End of module

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	48	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	591	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	0	0	252	00:00.2
\$255\$DUA28:[RUNOFF.SRC]DSRLIB.L32;1	1248	94	7	86	00:00.3

STC
V04-000

Processes the .SEND TOC command
STC --

C 8
16-Sep-1984 01:48:37
14-Sep-1984 13:08:11

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[RUNOFF.SRC]STC.BLI;1
Page 11
(4)

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:STC/OBJ=OBJ\$:STC MSRC\$:STC/UPDATE=(ENH\$:STC)

; Size: 591 code + 48 data bytes
; Run Time: 00:15.1
; Elapsed Time: 00:30.0
; Lines/CPU Min: 1656
; Lexemes/CPU-Min: 18063
; Memory Used: 141 pages
; Compilation Complete

0349

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY